Beyond Black-Box Optimization

A Review of Selective Pressures for Evolutionary Robotics

Stéphane Doncieux[1,2] **• Jean-Baptiste Mouret**[1,2]
{doncieux, mouret}@isir.upmc.fr

Doncieux, S. and Mouret, J.-B., "Beyond black-box optimization: are view of selective pressures for evolutionary robotics" (2014), *Evolutionary Intelligence*, DOI: 10.1007/s12065-014-0110-x, Springer Berlin Heidelberg, pp 1-23. The final publication is available at link.springer.com

Evolutionary robotics is often viewed as the application of a family of black-box optimization algorithms – evolutionary algorithms – to the design of robots, or parts of robots. When considering evolutionary robotics as black-box optimization, the selective pressure is mainly driven by a user-defined, black-box fitness function, and a domain-independent selection procedure. However, most evolutionary robotics experiments face similar challenges in similar setups: the selective pressure, and, in particular, the fitness function, is not a pure user-defined black box. The present review shows that, because evolutionary robotics experiments share common features, selective pressures for evolutionary robotics are a subject of research on their own. The literature has been split into two categories: goal refiners, aimed at changing the definition of a good solution, and process helpers, designed to help the search process. Two subcategories are further considered: task-specific approaches, which require knowledge on how to solve the task and task-agnostic ones, which do not need it. Besides highlighting the diversity of the approaches and their respective goals, the present review shows that many task-agnostic process helpers have been proposed during the last years, thus bringing us closer to the goal of a fully automated robot behavior design process.

1 Introduction

DESPITE decades of research of in robotics [164], even the most advanced robots are a far cry from the efficiency, adaptivity and, overall sophistication of animals. Bio-inspired robots import some ideas from these natural wonders [118–120, 152, 153, 60], with the hope of taking advantage of billion years of evolution. Evolutionary Robotics (ER) [141, 60, 53, 20] follows a close but different path: instead of trying to replicate the result of evolution, why not try to replicate evolution itself? Evolutionary robotics hence proposes to employ evolution-inspired algorithms to design robots or, more often, control systems for robots.

From the embodied cognition point of view [152, 153, 20], evolutionary robotics could lead to machines with their own vision of the world, devoid of anthropocentric bias. For instance, many mobile robots see the world as a colorless, two-dimensional world, because they perceive it through a LIDAR [164]; what is it like to think and act in such a world? Answering such a question is very challenging for humans, who experience a much richer world.

From the engineering point of view, evolutionary robotics aims to propose an automated engineering process [113, 53, 20], that is, a process in which engineers write specifications and a computer takes care of the design.

[1]Sorbonne Universités, UPMC Univ Paris 06, UMR 722, ISIR, F-75005, Paris, France
[2]CNRS, UMR 7222, ISIR, F-75005, Paris, France

Evolutionary Algorithms (EA) (see e.g. [46, 56, 43]) provide the algorithmic foundation of evolutionary robotics. In their modern form, these population-based optimization algorithms are composed of four components: a genotype, a genotype to phenotype mapping, a set of variation operators, and a user-defined function to be optimized, called a fitness function. This fitness function is always left to the user. Because no assumptions are made about the fitness function, evolutionary algorithms are often classified as "black-box optimization algorithms".

Viewing evolutionary algorithms as black-box optimization tools is seductive because optimization is a well-defined field of applied mathematics, and because black-box optimization can be used in many real-world situations. However, it has a side effect: it incentivizes researchers to work on what is *not* user-defined – the encoding and the evolutionary operators. As a result, evolutionary robotics focused for a long time on how to encode the morphology and the brain of robots (e.g., [165, 114, 85]) or how to encode neural networks (e.g., [95, 123, 171, 50, 59, 128, 169, 35]).

At any rate, however good the encoding is, crafting a fitness function is "notoriously difficult" [20, 134]. A first challenge is that evolutionary algorithms – like all optimization algorithms – do not possess any common sense: they exploit every way to maximize the fitness function, in particular those that take unforeseen shortcuts. For instance, let us imagine we want to evolve a neural network that would allow a mobile robot to avoid obstacles [141]. A straightforward fitness function simulates robots for some time in a simulator, and counts how many seconds they move without hitting a wall. The result will probably be disappointing: with such a fitness function, the robot usually does not move at all but, counter-intuitively, receives the maximum fitness score. A robot that does not move, after all, will not hit any walls! If the fitness function is improved so that the robot is forced to move, then we can expect evolved robots to move in a circle instead of exploring the environment. As illustrated by this example, a long refinement process is often required before obtaining a fitness function that unambiguously reflects the target behaviors.

A good fitness function is even more challenging to craft because it serves two different purposes: it both defines the goal and guides the search. Mixing up these two purposes makes sense in a black-box optimization because the fitness function is the only information that is available to identify promising solutions. This strategy works well when finding good solutions does not impose large detours that are not directly identifiable with the fitness function. Recent experiments, however, exhibited several tasks in which this kind of objective-based search was especially ineffective [106, 185]. In nature, fossils provides many examples of detours that would have been hard to find using objective-based search: many traits of animals and plants have been exapted, that is, they have been co-opted for a purpose for which they were not initially selected [76].

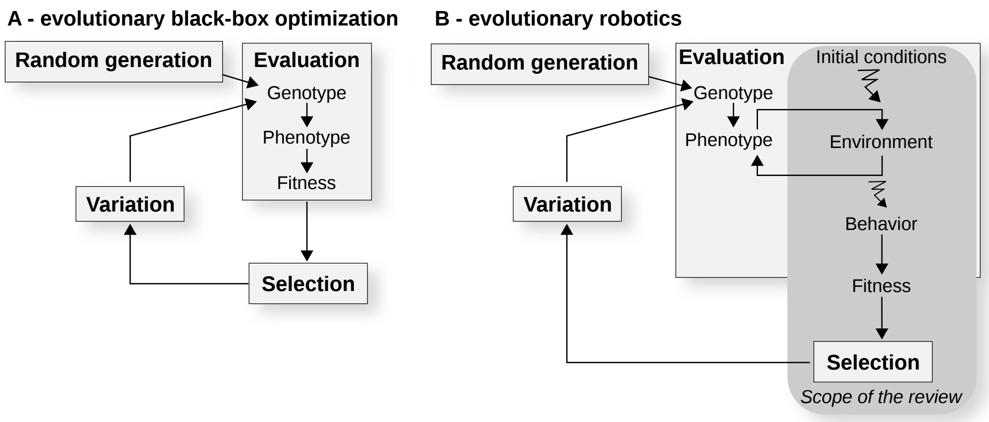

Fig. 1. (A) General principle of evolutionary algorithms. (B) Principles of evolutionary robotics. The dark gray area corresponds to selective pressures as reviewed in this paper.

Classic examples are bird feathers, which may initially have evolved for temperature regulation, and vertebrate bones, which may have been selected to store phosphates [76]. Similarly, the history of science and technology is full of critical but serendipitous discoveries. For instance, the concept of heating food by microwaves was discovered when working on radar tubes [157], and the effects of penicillin on microbes were first observed in a failed experiment about lysozyme. Because evolutionary robotics aims at creating artifacts as complex as life forms, it is reasonable to expect that such detours will have to occur in artificial evolution.

From the point of view of black-box optimization, problems when crafting fitness functions stem from users who do not specify what they are looking for with enough accuracy, and who do not provide the "right" heuristic to guide the search. Put differently, these are issues with the users, not with the algorithms. Such a view does not give evolutionary robotics much hope: if programming a fitness function to evolve simple behaviors is not straightforward, how could we hope to employ evolution to design vastly more complex and hard to define behaviors like, for instance, "being intelligent"?

Fortunately, *evolutionary robotics is not black-box optimization*: most experiments have both common challenges and similar setups. For example, most evolutionary robotics experiments involve testing robots, observing their behavior and attributing the fitness score (fig.1). Instead of only looking at the fitness score, designers of algorithms for evolutionary robotics can assume that the concept of behaviors exists and can be exploited. For instance, some recent algorithms compare behaviors to prevent the algorithm from converging toward a single family of behaviors [131], or to favor behaviors that have not been seen before [106]. The stimulating results achieved with these algorithms suggest that studying *selective pressures* may be at least as important as studying encodings and evolutionary operators [131].

Interestingly, the study of selective pressures is at the center of many, if not most, papers about biological evolution, whereas it has only recently been identified as a main topic in evolutionary robotics. Most of the early papers related to selective pressures are "guides" to help researchers design a working experiment, often by incorporating task-specific knowledge into the fitness function. For instance, some papers advocate the use of an incremental approach according to which the "practitioner" splits the task into sub-tasks and solves each of them separately [81, 40, 95, 132]; some other papers describe how to reward the achievement of intermediate useful behaviors [179]; many of them also discuss the use of noise in the fitness function, in particular to discourage over-specialized solutions [88, 89].

Two scientific advances have enabled the evolutionary robotics community to study selective pressures in a more generic way than fitness writing guides. First, multi-objective evolutionary algorithms (see, e.g. [46]) have demonstrated that ranking candidate solutions can be achieved in several ways, and not only by using a single fitness value for each individual. These algorithms have allowed researchers to stop tuning weights of complex, aggregated fitness functions and thus to focus on the content of the objectives [46]. They have also paved the way to helper objectives, which are adjunct objectives used to improve the performance of an evolutionary process [92, 90]. The second advance is Novelty Search [104, 106], which has demonstrated that guiding evolution with objective functions is not the only possibility. These two scientific advances have led several teams to call into question the dogma of a purely user-defined fitness function to guide an evolutionary algorithm. These two lines of work have thus renewed interest in some of the most fundamental questions of evolutionary robotics, like: what should be the driver of an artificial evolutionary process? Is the evolutionary process necessarily driven by a task-performance criterion? or what are the alternatives to performance objectives?

Overall, there are now dozens of papers in evolutionary robotics that are explicitly focused on selective pressures. Modifications of fitness functions have, however, always been present in the evolutionary robotics literature (e.g., fitness shaping or incremental evolution). The goal of the present paper is to analyze all these selective pressure modifications in a common framework.

Previous work on fitness functions for evolutionary robotics focused on the amount of prior knowledge included in the fitness function [134, 141]. Hence, Nolfi and Floreano proposed a classification of fitness functions with respect to three dimensions: explicit/implicit (measuring the way the goal is achieved versus measuring the level of attainment of the goal), external/internal (measuring fitness through an external observer versus measuring it internally with the robot), and functional/behavioral (rewarding a particular working modality versus the quality of the behavior)[141]. Nelson et al.[134] focused on a single axis that represents the amount of a priori knowledge incorporated in the fitness function. Both classifications rely on the same reasoning: exploiting prior knowledge helps ER to find solutions quickly, but it prevents discovering original solutions; to make fair comparisons between approaches, therefore, both the performance and the level of autonomy of the evolutionary process must always be taken into account. Nevertheless, experiments with Novelty Search show that prior knowledge can be misleading [104, 106]. In addition, the recent literature contains many examples of fitness modifications that do not depend on the targeted task, that is, modifications that cannot be distinguished on a “prior knowledge” axis. Last, prior knowledge is difficult to quantify precisely.

The present review is focused on the issues addressed by modifying the fitness function (why?) and on the techniques proposed (how?). We divide selective pressures modifiers into *goal refiners* and *process helpers*. *Goal refiners alter the search space* so that solutions with classic issues are avoided. Consequently, they change the maximum of the fitness function. For instance, goal refiners have been proposed to avoid behaviors that work in simulation but not on the real robot [98], or to improve the reactivity of evolved controllers [103]. *Process helpers alter the search process*, most of the time by changing the method used to identify the most promising solutions. For example, process helpers can mitigate premature convergence by encouraging behavioral diversity [131], or guide the process by providing intermediate goals, like in many incremental evolution experiments [81, 40, 95]. Goal refiners and process helpers can both be task-specific, that is they can include knowledge on how to reach the goal, or task-agnostic, that is, the same code can be used for several tasks.

We first describe the specificities of evolutionary robotics and therefore what generic knowledge can be exploited by algorithms. We then identify the main challenges of evolutionary robotics. The classification of techniques found in the literature is then performed, first by the kind of approach (e.g., goal refiner or process helper), the status with respect to the task (specific or agnostic), the challenges it addresses and lastly the family of approaches it belongs to (e.g. multi-objective optimization).

2 What is common to evolutionary robotics experiments?

2.1 Common features

A robot is a system that receives information from its environment. It can move and modify the environment through its actions. It exhibits particular dynamics influenced (or not) by its current state, by some control outputs u, and by external factors e like, for instance, environment conditions or the actions of other robots. Its dynamics can be modeled with a differential equation as follows:

$$\dot{s} = G(s, u, e) \tag{1}$$

where s denotes the state of the robot and where $G(.)$ models the physical laws governing the interaction between the robot and its environment.

As a first approximation and to simplify the model, this equation can be expressed in discrete time as follows:

$$s(t+1) = G(s(t), u(t), e(t)) \tag{2}$$

Designing a robot behavior through evolutionary algorithms means looking for u[1] to reach trajectories of the system that have desirable features. The evolutionary process relies on one or more fitness objectives f_i evaluating the performance of a genotype g. These fitness objectives will depend on the system's trajectory:

$$f_i(g) = F_i(s_0^{(i)}, s^{(i)}(1), ..., s^{(i)}(T^{(i)}), x^{(i)}) \tag{3}$$

where $T^{(i)}$ is the evaluation length associated with f_i, $x^{(i)}$ represents other factors that the fitness objective may depend on, and $s_0^{(i)}$ is the initial state of the robot when starting the evaluation of f_i. $s_0^{(i)}$ is a parameter of this evaluation. $s^{(i)}(1), ..., s^{(i)}(T^{(i)})$ are iteratively computed with equation 2.

To sum up, every ER experiment requires evaluating the behavior of a robot once or several times. Besides u and G, each evaluation i relies on:

- $s_0^{(i)}$: the initial state;
- $T^{(i)}$: the evaluation length;
- $e^{(i)}$: the external conditions.

Exploiting any of these features makes an ER algorithm leave the category of black-box optimization algorithms (fig. 1).

[1]At this modeling level, it can be hypothesized that the morphology can be included in u.

2.2 Specific challenges

2.2.1 Premature convergence

The search space explored by a typical ER experiment is large and even unbounded, in particular when evolving neural network structures. The evaluation of a solution results from the observation of a dynamical system. As for any dynamical system, a small change in the parameters may result in a bifurcation [11], i.e., in a sudden and drastic change of behavior. When evolving robots, a small change in the controller parameters may make the robot collide with some obstacles and thus completely change its behavior. Likewise, a robot engaged in a locomotion task may fall or not. Bifurcations are thus not rare when evolving robots and create discontinuities in the fitness values. Fitness plateaus are common in ER [166]. Typical ER fitness landscapes are then large, at least partly rugged and include plateaus. They are not easy to explore, which results in a clear symptom: the search often gets trapped in local optima. The generated solutions do not satisfy the expectations of the user, even if the search is allowed to go on for a large number of generations. We will refer to this phenomena as the *premature convergence challenge* [68, 56]. It has also been called the bootstrap problem [129].

Another phenomenon can actually explain premature convergence. The fitness function has two different roles: defining the goal and guiding the search. A fitness function may well describe what is expected, but it may also drive the process in the wrong direction. Such a fitness function is called *deceptive*. Lehman and Stanley argue that most, if not all, goal oriented fitness functions exhibit such deceptive properties and that they should thus not solely be taken into account during the search [106].

Premature convergence may thus be due to many different factors such as the lack of gradient, a deficient exploration or a deceptive fitness function. The challenge of overcoming this problem is not specific to ER and generic solutions have been proposed (see [56] for a review). This problem is a critical challenge in ER and this article will focus only on solutions that are specific to ER or that have been tested on an ER experiment.

2.2.2 Fitness definition

How can we quantitatively describe the behavior that is expected to emerge from the evolutionary process? This question may, in certain cases be particularly difficult to answer. Even in simple cases, defining a fitness function leading to an expected behavior is challenging. To design a robot that avoids obstacles, simply minimizing the number of collisions is not sufficient as it will generate robots that do not move at all. Likewise, if the robot is forced or encouraged to move, the risk is that it blindly follows a circular trajectory in a place where there is no obstacle. The behavior is then the one expected, but if the robot is put in front of an obstacle, it will not be able to avoid it. In this case, the evaluation process is not an appropriate way to check the desired property and there are unfortunately no theoretical tools nor frameworks to guide this tedious trial-and-error evaluation design process. Furthermore, designing an adapted evaluation process requires technical skills. It would be interesting therefore if an autonomous behavior design method could remove these needs so that non experts, like children, could use it [116].

Living creatures subject to natural selection have no “goal” other than transmitting their genes. The selective pressures exerted on them will depend on their ecological niche, which is local and may change over time. In light of the open-ended property of natural evolution, we may question the validity of driving artificial evolution mainly by maximizing a constant task-based objective function.

The *fitness definition* challenge will refer to the problem of designing a fitness function together with the conditions of the evaluations, to reach some expected behavior. Works that replaces any need for an analytical function to evaluate the performance of the solutions being tested, and thus bypass this problem will also be considered as addressing this challenge.

Formally, this challenge corresponds to the design of f and everything it depends on, i.e. F, s_0, T and e.

2.2.3 Reducing evaluations

The natural selection process took several billions of years to create complex creatures like humans. Even the simplest multi-cellular creatures have required billions of years to appear. Algorithms inspired from natural selection are also slow as they require evaluating the performance of a large number of potential solutions. Finding how to reduce the time devoted to evaluations is thus a critical issue, in particular when robots with complex morphologies and behaviors are searched for. This problem can be tackled from two different and complementary points of view: either by trying to reduce the number of evaluations or by trying to reduce the evaluation length T (which may vary from one evaluation to another). Both aspects will be grouped together into a single challenge: *reducing evaluations.*

2.2.4 Reality gap

Evolutionary robotics experiments can be run directly on real robots [61], but the required number of evaluations and the risk of damaging the robots encourages minimizing evaluations on real robots. The availability of fast simulators like ODE[2] or Bullet[3] has allowed ER researchers to rely, at least partly, on simulations. The advantages are numerous: simulations are generally faster than real time; they allow a parallelization of evaluations, which is particularly interesting when using modern clusters; and all problems related to repeating a robotic experiment a large number of times are avoided (mechanical fatigue, motor or sensor failures, etc). When the target is a real robotic platform, the inevitable discrepancies between the simulated robot and the real one introduce a new problem: controllers generated in simulations will be adapted to the simulation but not necessarily to the real robot. If they exploit a feature that is specific to the simulation, the behavior on the real robot will be less effective or maybe completely ineffective, thus leading to the *reality gap* problem [89, 97, 98].

In the proposed formalism, this corresponds to situations in which G changes. If G_s describes the behavior of the simulated robot and G_r the behavior of the real robot and $s_s(t)$ and $s_r(t)$ the respective corresponding states, the problem consists of ensuring that the difference of fitness between the two situations remains, as much as possible and at least locally, consistent:

$$F(s_s^{(1)}(0),\dots,s_s^{(1)}(T)),x > F(s_s^{(2)}(0),\dots,s_s^{(2)}(T),x)$$
$$\Rightarrow F(s_r^{(1)}(0),\dots,s_r^{(1)}(T),x) > F(s_r^{(2)}(0),\dots,s_r^{(2)}(T),x)$$

Furthermore, for practical reasons, it is interesting that the difference between the fitness values associated with the same genotype in simulation and in reality remains bounded and as small as possible:

$$|F(s_s(0),\dots,s_s(T),x) - F(s_r(0),\dots,s_r(T),x)| < \epsilon$$

There may be different ways to address this challenge. We will focus here on algorithms and methods that change the selective pressure.

[2] http://www.ode.org/
[3] http://bulletphysics.org/wordpress/

2.2.5 Generalization

During an ER experiment, the potential solutions are evaluated on a set of evaluations defined by an initial state $s_0^{(i)}$, a finite evaluation length $T^{(i)}$ and external conditions $e^{(i)}$. Consequently, only a limited number of different situations will be encountered by the robot during an evaluation and thus taken into account in the fitness. A solution optimizing the fitness meets the expectations in these situations, but nothing can be said for other situations and performance drops are often observed [48]. If ER is to be used in real and practical situations, end users will expect the evolved behavior to be robust to variations in the environment. Any evolved controller whose behavior will be specific to the initial conditions and the particular environment used during evolution will be useless in practice. Furthermore, as $T^{(i)}$ is a critical factor with regard to the duration of an experiment, it is generally chosen to be as short as possible. The challenge is then to define methods to generate a controller with only few evaluations while ensuring that it is successful in different and new contexts [154]. This will be called the *generalization* challenge. This issue is not specific to ER and holds for many machine learning algorithms [1], but we will focus here only on methods that (1) have been applied to ER experiments and (2) rely on selective pressure adaptation.

3 How to influence selective pressures?

Evolutionary algorithms rely on the Darwinian principle of variation and selection of the fittest. Any aspect that may influence this selection process is referred to as a selective pressure. In the following, different categories of approaches aimed at influencing the selective pressures are presented. It should be noted that these approaches are not exclusive and can, for some of them at least, be combined.

Mono-objective EA In mono-objective evolutionary algorithms, a single scalar fitness function is used to drive the evolutionary search process. This approach corresponds to the most classical EA. Genetic algorithms [83], evolution strategies [162], evolutionary programming [66] and genetic programming [99] were all mono-objective EAs when they were first proposed.

Multi-objective EA While mono-objective EAs aim to find the optimal solution of a unique function, multi-objective EAs are designed to generate a set of optimal trade-offs between several objectives [46]. Trade-offs are optimal with respect to ordering relations specifically designed for multi-objective spaces, often the Pareto dominance relation, defined as follows:

Definition 3.1 (Pareto dominance.) *A solution x^* is said to dominate another solution x, if both conditions 1 and 2 are true:*

1. *the solution x^* is not worse than x with respect to all objectives;*
2. *the solution x^* is strictly better than x with respect to at least one objective.*

This dominance relation is not a strict ordering. This is why multiple trade-off solutions exist: some solutions can have very different objective values and yet neither dominate nor be dominated one by the other. Multi-objective problems can be turned into mono-objective problems with an appropriate aggregating function – like a weighted sum, for instance. Aggregating functions require parameters – e.g. objective weights – or some knowledge about the objective space – e.g., the extremum values of each objective. One advantage of multi-objective algorithms is that they do not need such parameters. Another advantage is that, as the search will advance along a front of non-dominated directions instead of along a single direction, it can lead to a better convergence rate [93].

Coevolution Coevolutionary algorithms are EAs in which the fitness of a particular individual depends on other individuals, which are also evolved [150]. These approaches are closer to what happens in the living world, where the selection process depends on the ecological niche of a particular species, including other evolving species (predators or prey, for instance). This leads to fitness functions that are relative [2], and that may be based on competition or on cooperation, within the same species or between different species.

***Ad hoc* EAs** Most work on selective pressures uses a standard EA and investigates the modification of one of its components (the fitness, the selection operator, the ranking strategy, etc.); some papers, however, propose modifications of the evolutionary loop itself, for instance by alternating between two independent EAs. These papers propose new EAs motivated by ER needs. They will be assigned the label "*Ad hoc* EAs".

Evaluation conditions The evaluation of the fitness objectives relies on the initial state of the robot ($s_0^{(i)}$), on the evaluation length ($T^{(i)}$) and on external conditions ($e^{(i)}$). Any modification or adaptation of these has an impact on the fitness values and on the selection process. An approach that proposes to modify any of these aspects will be assigned the "evaluation conditions" label.

Fitness shaping The fitness objectives are critical in driving the selection process. The selection algorithm will mostly rely on these objectives to decide which individual will survive or be used as a genitor of new individuals. Besides the most straightforward description of the expected robot behavior, new terms refining these properties can be added to the fitness objectives in order to avoid undesired behaviors – e.g. avoiding obstacle by standing still – or to help the search – e.g. walking on its legs requires making legs move. This process will be referred as "fitness shaping", which corresponds to modifications of $f_i(g)$.

Staged evolution An ER experiment in which several EA experiments are sequentially launched will be referred to as staged evolution. The best individuals of one particular EA run will feed the next one and successive EAs will rely on different selective pressures (typically different fitness functions or evaluation conditions).

Interactive evolution In a typical EA, individuals are evaluated on the basis of fitness objectives. These are analytic functions implemented in the EA to automatically evaluate each new individual. Interactive evolution consists of relying on evaluations made by humans [173], with the idea that human intuitions may be difficult to capture in a single and static analytic fitness function.

4 A classification

The fitness objectives classicly serves two different roles: defining the goal and guiding the search. Based on this assertion, the literature on selective pressures has been split in two different categories: goal refiners and process helpers (fig. 2).

Definition 4.1 (Goal refiner) *A goal refiner aims at changing the optimum (or optima) of the fitness function by adding new requirements.*

In the current literature, goal refiners mostly address the issues that stem from the reality gap (section 2.2.4), generalization (section 2.2.5) and fitness definition (section 2.2.2).

Jakobi's work on the reality gap [89, 88] is a typical example of a goal refiner. Jakobi realized that evolved neural networks critically relied on irrelevant details of the simulation, whereas he aimed to find more general and robust solutions. He therefore designed a strategy wherein solutions could not rely on such details: he added noise in the simulator, hiding details in an "envelope of noise". By doing so, he modified the optimum of the fitness function to avoid attractive optima that were not robust enough to work on the real robot. Put differently, he added a principled, general requirement that was not present in the initial formulation of the task, but which is implicit in many tasks.

Definition 4.2 (Process helper) *A process helper intends to increase the efficiency of the search process without changing the optimum(optima) of the fitness function*[4].

In the current literature, process helpers mostly address issues with premature convergence (section 2.2.1) and fitness definition (section 2.2.2). For instance, behavioral diversity [130, 129, 131] is a process helper: the diversity of the population is encouraged by adding an objective [46] that rewards the originality of each behavior with regard to the current population; such a diversity preservation aims at avoiding the premature convergence of the EA, that is, at improving the performance of the evolutionary process. This approach does not change the optimum of the fitness function because the diversity objective is discarded at the end of the evolutionary process and, as a result, final solutions are only ranked by their fitness value.

Goal refiners and process helpers can exploit some knowledge specific to the task or not. Each category is then further split in two subcategories: task-specific and task-agnostic.

Definition 4.3 (Task-specific) *Task-specific goal refiners/process helpers incorporate knowledge on how to solve the task.*

One of the main characteristics of task-specific approaches is that they cannot be transferred to other tasks without adaptations. Much of the early work on selective pressures is task-specific because it requires an analysis of the task by the experimenter. For instance, staged evolution [40, 81] proposes splitting the final task into several intermediate sub-tasks and solving each of them sequentially. When this split is not automatic, the quality of the results critically depends on the task and on the expertise of the experimenter.

Definition 4.4 (Task agnostic) *Task agnostic goal refiners/process helpers do not exploit knowledge about how to solve the task.*

In contrast to task-specific approaches, task-agnostic approaches can easily be transferred to other tasks with limited or even no modification at all. Behavioral diversity, for instance, is a task-agnostic helper because the same helper can be used for several related tasks. For example, a behavioral diversity objective based on the position of the robot at the end of each evaluation has been used for maze navigation [106, 131], biped locomotion [106, 110] and hexapod locomotion [51, 52]. No approach is, however, fully task-agnostic. For instance, the end position of one robot is irrelevant in a multi-robot setup, therefore the helper would have to be modified to take several robots into account. Likewise, in a ball-collecting task, the end position of the robot can be replaced by the end position of the balls to capture more precisely behavioral features [51, 131, 52].

4.1 Goal refiners

Goal refiners are listed in table 1 (an up-to-date version of this table is available online at: `http://pages.isir.upmc.fr/selective_pressures/`).

[4]Some process helpers may have side effects and change the optimum of the fitness function, whereas it was not the intent of its authors. They are here considered to be helper processes as long as such optimum modifications are not straightforward and have not been clearly identified.

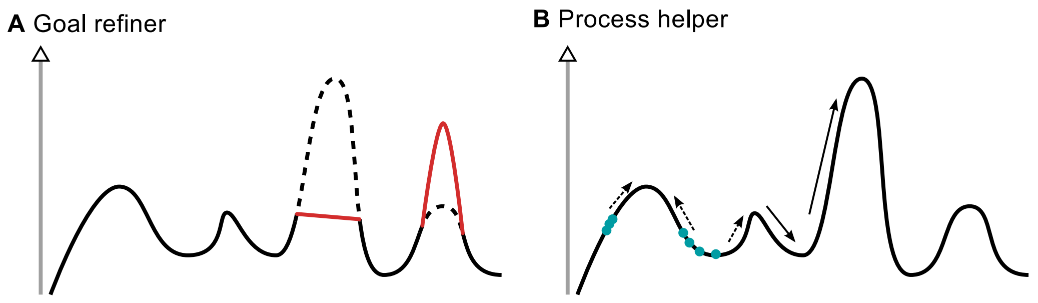

Fig. 2. Illustration of modifications of selective pressure, for a 1-dimensional function to be maximized. (A) Goal refiner: the optimal solutions are changed by adding new requirements. Goal refiners can remove fitness peaks and add new ones. (B) Process helpers: optimal solutions are not changed, but the search process is modified.

4.1.1 Task-specific

Behavioral consistency [146, 147] is a method for defining a selective pressure that consists of rewarding solutions that behave the same (or differently) in different scenarios. The goal is to force the robustness and generalization ability of generated solutions by ensuring that the corresponding behavior is the same in the presence of noise or distractors, for instance, [146]. Likewise, by encouraging exhibiting a similar behavior in different situations, it can reward the appearance of a circuit able to detect and memorize some states, i.e. a memory [147]. This approach was applied to a delayed response task in which a robot had to choose a branch to follow in a T-maze depending on a previously-received signal. The behavioral consistency relies on a dedicated objective to be optimized in a multi-objective context. It is considered to be task-specific because it requires defining several different scenarios for which the behavior should be similar or different. This approach requires expertise about the task. Behavioral consistency has been used to validate hypotheses on the impact of noise and occlusion on the emergence of internal representations [144] in a robot navigation task. A significant correlation was identified in this work between generalization ability and internal representation. It is thus considered here as addressing this challenge.

4.1.2 Task-agnostic

A significant number of studies can be attributed to this category. We have chosen to present them with regard to the challenge they address.

Reality gap As evolutionary algorithms require a large number of evaluations, they are often run, for practical reasons and at least partly, in simulation. Due to the opportunistic property of EAs, features specific to the simulation can be exploited, and generated solutions may thus not transfer to reality: this is the reality gap. This challenge has drawn a lot of attention with approaches aimed at modifying the features of generated solutions, i.e. goal refiners, so that generated solutions are effective on the real robot. We have regrouped the approaches tackling this challenge with selective pressures in three different categories: constant simulation, robot-in-the-loop and adaptive simulation.

Simulation-based approaches rely only on the simulation and adapt the algorithm so that solutions robust to the transfer between simulation and reality are found. In these approaches, the simulation is constant during the evolutionary experiment. Jakobi proposes to evaluate individuals in a minimal simulation [88]. As a simulation can hardly accurately model every single physical phenomenon, he proposes to build minimal simulations that accurately model only a selected subset of robot-environment interactions. Other aspects are hidden in an envelope of noise, so that no solution can exploit them. The approach was applied to a T-maze navigation task with a Khepera robot, and to a visual discrimination task on a gantry robot. Other authors also propose to add noise while evaluating a solution in order to reduce the reality gap [121, 75][5], for both a Khepera robot obstacle avoidance task and a double pole balancing task. Boeing and Braunl propose a different approach: instead of evaluating in a single simulation, solutions are evaluated in several different simulations at the same time [12]. The fitness is the normalized average value of the performance as measured in the set of available simulations. Coping with simulations variability is expected to promote the robustness of controllers. All these approaches define specific evaluation conditions – either with noise or with different simulations – in order to help crossing the reality gap. It was tested on a wall following task for an autonomous underwater robot. Lehman et al. propose a completely different approach. Their hypothesis is that a reactive agent will also be robust and will thus more easily cross the reality gap [103]. They propose to use mutual information to measure the statistical dependence between the magnitude of changes on a robot's sensors and effectors. An objective is thus defined and optimized in a multi-objective EA alongside other objectives to promote the reactivity of the generated controllers and the approach is applied to maze navigation tasks.

While still keeping a constant simulation, another approach consists of evaluating several solutions directly on the real robot [98, 97, 133, 142]. Relying on the hypothesis that reasonably good simulators do indeed exist, the approach proposes learning a model of behavior discrepancies between simulation and reality in order to avoid the most unrealistic behaviors. The evaluations on the real robot are used to learn a model of the transferability of a particular solution between simulation and reality. The transferability model predicts how similar a particular behavior will be between simulation and reality. This predicted transferability is used as a new objective in a multi-objective EA alongside other objectives so that generated solutions tend to behave the same in simulation and in reality. The approach has been applied to a quadruped [98, 97] and biped [142] locomotion tasks as well as to a T-maze navigation task [98]. As the number of evaluations on the real robot is reduced, this approach is considered to also address the reducing evaluation challenge.

With the hypothesis that the reality gap comes from discrepancies between simulation and reality, experiments on real robots can be used to design or improve simulations. In the following, the simulation model is no longer constant but is adapted on the fly or even learned from scratch: the evaluation conditions then change during a run. Most of this work relies on co-evolution or on *ad hoc* evolutionary algorithms that allow the evolution of both simulations and robot controllers. Bongard et al. propose an approach based on co-evolution, the Exploration-Estimation algorithm, in which a population of simulations co-evolves with a population of con-

[5]In these studies, a model of the robot is learned before launching the evolutionary algorithm. It was put in this category as, after the initial training – independent from the evolutionary algorithm –, the simulation model was not updated.

	why?					how?							
	Fitness definition	Reducing evaluations	Generalization	Premature convergence	Reality gap	Shaping	Multi-objective EA	Staged	Mono-objective EA	Evaluation conditions	*Ad hoc* EA	Co-evolution	Interactive EA
Task-specific													
Behavioral consistency [147, 146]			•	•			•						
Noise occlusion [144]			•						•	•			
Task-agnostic													
Back to reality [187–189]					•					•	•		
Breeding robotics [116, 115]	•								•				•
Co-evolution models/tests [44, 96]					•		•			•		•	
Embodied evolution [182]	•				•						•		
Empowerment [91]	•								•				
Enveloppe of noise [88]					•				•	•			
Fitness based on information theory [155, 167, 168, 47]	•								•				
GSL [57]					•					•	•		
Interactive evolution [78, 137, 54]	•								•				•
MONEE [79]	•										•		
Model-based neuroevolution [75]					•				•	•			
Multiple simulators [12]					•				•	•			
NA-IEC [186]	•								•				•
Novelty search w. local comp. [37, 107]	•			•			•						
ProGAb [154]		•	•				•						
Reactivity [103]	•		•		•		•						
Sampling & noise [121]					•				•				
Self-modeling [15, 26, 23–25, 13]					•						•	•	
Transferability [97, 142, 98, 133]		•			•		•						
mEDEA [27]	•										•		

Table 1. Goal refiners. Each line corresponds to an article or set of articles about a similar topic (with respect to selective pressures). The first five columns describe the addressed challenges and the remaining ones the way they have been addressed.

trollers [25]. It has been used in particular for damage recovery on quadruped and hexapod robots [23]. Simultaneously, Zagal et al. proposed the *back-to-reality* algorithm [189, 188, 187], a similar approach that consists in performing an optimization in simulation, transferring some selected solutions to reality and exploiting the corresponding data to improve the simulation before optimizing in simulation again. These different steps are repeated until the behavioral requirements are met. The approach has been used for a locomotion task on a quadruped robot [189, 188] and on a humanoid robot [187]. Farchy et al. propose a similar approach with a choice made by the experimenter on which parameters to focus on for the next optimization [57]. This approach was applied to a Nao humanoid robot locomotion task. Bongard et al. propose an extension of the co-evolution approach in which actions are explicitly sought to challenge current candidate models [26]. It was later shown that, besides model disagreement, looking for actions avoiding bifurcations is important for generating reliable models [13]. Extensions to multiple robot were done in [15]. Koos et al. propose a similar co-evolution approach, but implemented in a multi-objective EA [96]: models are evaluated on their ability to reproduce observed data, and controllers are evaluated with three objectives: their ability to discriminate between models, how close they are to the desired behavior, and how stable they are. The stability is evaluated as the variance of behaviors observed between slightly mutated versions of the controller. The approach was applied to trajectory following tasks on a quadrotor. Embodied evolution goes further and proposes to rely only on real robots [182]. In this case, the reality gap no longer exists and the decentralized features of embodied evolution allows the parallelization of the approach and the scaling to a large number of robots.

Generalization A solution optimizing a fitness function ensures that the corresponding behavior matches the expectations in the contexts used for evaluation. Some methods try to ensure that generated solutions will meet these expectations in new and unforeseen contexts. The reality gap can be considered as a special case of the generalization challenge: the solutions should meet the expectations in reality after being evolved in simulation. Much less work has been devoted to this challenge, although some of the work on the reality gap can be applied to the generalization challenge. This is the case of the previously mentioned work of Lehman et al. on reactivity [103]. Several authors propose methods based on coevolution with the idea of having evaluation conditions that are automatically adapted to the performance of current solutions. Berlanga et al. propose "Uniform Coevolution", a method of evolving the weights of a neural network controller and the evaluation conditions simultaneously (more precisely the initial state $s_0^{(i)}$) [10, 9]. This method was applied to a robot navigation task.

For a similar application, Sakamoto and Zhao likewise use coevolution and compare it to incremental evolution in which new conditions are incrementally added to the evaluation process [158]. Coevolution was revealed to be a better solution, provided that it is exploited in the right manner. Pinville et al. propose a different approach based on the assumption that testing the generalization ability is time-consuming as it requires performing multiple evaluations. Inspired by the transferability approach [98], they propose learning a surrogate model of how a behavior will generalize to new evaluation conditions [154]. Several solutions are tested on a large set of conditions, thus better evaluating their generalization ability. All solutions are tested on a limited set of conditions and a surrogate model is built in order to predict, out of the behavior on the limited set of conditions, to what extent the corresponding behavior will generalize. This is used as a new objective to be optimized in a multi-objective EA alongside other objectives, and tested on ball-collecting and T-maze tasks. As for the transferability approach, the number of evaluations is reduced thanks to the surrogate model. This approach is thus considered to also address the reducing evaluation challenge.

Fitness definition Transforming expectations of the robot behavior into an analytical function that can evaluate generated solutions is often a difficult task. In some approaches, no explicit, goal-directed fitness function is used: the selection pressure is applied by other means. We distinguish three families of approaches: (1) interactive evolution, (2) information theoretic approaches, and (3) implicit fitness functions, in which the selection emerges from the interaction between the agents and their environment.

Interactive evolution relies on humans to estimate the performance of solutions. It has been used, for instance, to design pictures [163] or 3D objects [34]. Interactive evolution relaxes the expertise required in creating or programming a robot, and it even allows children to program robots [116, 115]. Gruau and Quatramaran used interactive evolution in conjunction with cellular encoding to design an octopod walking controller [78] and Nojima et al. use it for robot hand trajectory generation [137]. Evolutionary algorithms require numerous evaluations which, when performed by a human, may result in significant fatigue that can impede the performance of the search. In a robot behavior design experiment, Dozier proposed to learn a model of user preferences and to use it for further evolution, thus reducing human fatigue for a Khepera navigation task [54]. Woolley and Stanley proposed to associate it with novelty search (section 4.2.2) to exploit both the searching ability of novelty search and human insights on potential stepping stones. They demonstrated the technique in the deceptive maze navigation domain [186]. Interactive evolution has also been used to help mitigate premature convergence (see section 2.2.1, paragraph "semi-interactive evolution")[22, 31].

Fitness functions based on Shannon's theory of information have been investigated by several authors because they may provide a task-independent way to evaluate the "interestingness" of a behavior. Such fitness functions rely on the assumption that interesting behaviors are those that correspond to rich experiences in the environment, which should translate to high-entropy sensory-motor streams. These approaches are related to Novelty Search (section 4.2.2) because both approaches aim to maximize interestingness; however, they historically differ in their goal: information theoretic approaches aim at proposing a task-independent fitness function, whereas Novelty Search is more designed to mitigate deception. In addition, information theoretic approaches are individual-centered, because the interestingness of an individual does not depend on the other solutions, whereas Novelty search is process-centered, because the interestingness of an individual depends on what has already been discovered by the evolutionary process. Among those who investigated fitness based on information theory, Sporns and Lungarella [168] showed that the maximization of the information structure of the sensory states experienced by embodied and situated agents can lead to the development of useful behavioral skills in a simplified virtual agent, like the ability to foveate and to touch a moving object. Klyubin et al. [91] focused on the information contained in the sensory stream, because "the more of the information about the sequence of actions can be made to appear in the sensor, the more control or influence the agent has over its sensor". They propose a utility function called "empowerment", defined as the information-theoretic capacity of an agent's actuation channel, and show how maximizing empowerment influences the evolution of both sensors and actuators. In a less abstract setup, Prokopenko et al. [155] showed that fast locomotion of a snake-like, simulated robot can be achieved by maximizing the generalized correlation entropy (a lower bound of Kolmogorov-Sinai entropy) computed over a multivariate time series of the actuatorsâĂŹ states. In collective robotics, Sperati et al. [167] showed that mutual information in state and time between the motor states of wheeled robots leads to the evolution of various coordinated behaviors. Last, Delarboulas et al. [47] combined ideas from information theory with an on-board (1+1)-ES and compared their approach to Novelty Search in a maze navigation task. They conclude that both approaches have their merits.

No explicit fitness function drives natural species' selection process. Being able to propagate its genes implies surviving long enough to reproduce [38], which in turn implies having the required skills to reach this goal in the ecological niche the creature lives in. This metaphor can be directly imported into ER without the need to make the fitness function explicit. Bredeche and Montanier propose mEDEA, a minimal Environment-Driven Evolutionary Algorithm, to implement these ideas in an ER context [27]. In this approach, genomes are broadcast to every encountered robot. The genome controlling a robot is periodically changed and drawn from mutated versions of the genomes that the robot received. To survive, robots should navigate and find food items. Haasdijk et al. build upon this idea and propose MONEE (Multi-Objective aNd open-Ended Evolution), an implementation of these principles in which the environment-driven pressure is associated with a task based pressure [79]. This pressure is built on the value of credits that a robot can amass during its lifetime while performing tasks (collecting pucks of different colors). Each task is associated with a particular kind of credit and a market mechanism is defined to avoid robots concentrating on the easiest task. The more credits a genome has, the higher its chance of being selected. [109] follow a different perspective. In the living world, all living creatures do not compete together. The competition is local to a niche. Letting niches appear and accumulate increased evolvability [109]. A search simulating niches and local competition was implemented by associating novelty search with local competition for generating diverse morphologies and controllers of robots [107].

4.2 Process helpers

The vast majority of process helpers aim to mitigate premature convergence (table 2, an up-to-date version of this table is available online at: `http://pages.isir.upmc.fr/selective_pressures/`), that is why they are analyzed here with respect to the technique investigated and the results obtained.

4.2.1 Task specific

Incremental evolution Several early pieces of work in ER showed that many tasks are too hard to be solved with a basic EA and a high-level fitness function [81, 180, 74, 127]. For instance, Urzelai et al. [180] found it hard to evolve a light-seeking behavior in an arena without having previously evolved an obstacle-avoidance reflex [180]. From a selective pressure point of view, this issue is an instance of premature convergence (section 2.2.1). From an engineering point of view, one solution is to break down the problem into simpler sub-problems that can be solved sequentially. Following this classic methodology in engineering, many authors investigated incremental evolution processes in which evolution occurs in stages. The target task is split into ordered sub-tasks and a fitness function is designed for each sub-task. The population is first evolved using the first fitness function. After a user-defined number of generations or if the population has reached a sufficient performance level, the fitness function is replaced by the one that corresponds to the following sub-task. Task splitting is produced by the experimenter after having analyzed the task and potentially some preliminary experiments. The change of selective pressure is therefore task-specific.

De Garis is, to our knowledge, the first author to have employed staged evolution for an evolutionary robotics experiment [40]; he called his process "behavioral memory". In his experiments, he evolved a neuro-controller for a simulated "walking stick biped" using three successive fitness functions: (1) moving the legs in a "step-like" motion, (2) making as many steps as possible and (3) covering the maximum distance. Harvey et al.'s work on the Gantry robot [81] followed a similar approach but for a different task: tracking a target. Three stages were used, from locating a large immobile target to tracking a smaller, moving one. Parker [151] followed a similar incremental strategy to evolve gaits for a hexapod robot; Barlow et al. [8] did the same for controllers of simulated UAV, but using a MOEA instead of a mono-objective EA; Barate and Mazanera [7] employed two phases to evolve vision algorithms for mobile robots, where the first phase was based on behavior imitation and the second one on goal-reaching evaluations.

Instead of replacing the fitness function with a new one at each stage, some authors propose to keep the same fitness function but gradually modify the evaluation conditions, usually from simple conditions to challenging ones. These authors called these approaches "behavioral complexification"[74, 128], "behavior chaining" [14], "dynamic scaffolding"[18] or "incremental shaping" [3, 18]. Gomez et al. [74] thus worked on a prey-capture task that was parameterized with the prey speed and the delay before starting the pursuit; they defined ten ordered sub-tasks of increasing difficulty. Bongard et al. investigated automatic difficulty tuning with a task in which a simulated legged robot has to grab an object and lift it [14, 3, 19, 18]. They proposed two different algorithms: a hill climber in which difficulty is decreased when too many individuals fail and increased when they often succeed [14, 3], and a variant of the Age-Layered Population Structure (ALPS) algorithm [86, 84, 21, 18]. Bongard et al. highlighted that the order in which behaviors are evolved critically impacts the performance of the evolutionary process [14, 3]. They also noted that starting with changes in the morphology (morphological scaffolding) can synergize with changes in the environment (environmental scaffolding), but only if they occur in this specific order [19].

Another popular variant of staged evolution is to use the best result of stage $N-1$ as a component to build the candidate solutions of stage N, whereas many staged evolution processes keep the same population and only change the fitness function. This variant of staged evolution is sometimes referred as "modular decomposition" [180], "behavioral decomposition" [128] or "hierarchical evolution" [55]. When evolving a controller for a hexapod robot, Lewis et al. [111] first evolved a neural oscillator. In the second stage, they evolved a network of neural oscillators to allow the robot to walk. Also working on a hexapod robot, Kodjabachian et al. [95] first evolved a walking neuro-controller. In a second stage, they copied the best walking controller to each individual of a new population, froze it, and evolved a second neuro-controller, connected to the walking controller, to allow the robot to follow a gradient (e.g. an odor). In the last stage, they evolved a third neuro-controller, also connected to the walking controller, to allow the robot to avoid obstacles. Urzelai et al. employed a similar approach [180] to evolve neuro-controllers for a Khepera robot that has to move in an arena, avoid obstacles and periodically recharge its battery at the recharging station. De Nardi [45] also relied on a behavioral decomposition process to evolve a neuro-controller for a helicopter. Duarte et al. [55] divided a "robotics rescue" task into three different sub-tasks: (1) exit the room and enter the maze, (2) solve the maze to find the teammate, and (3) guide the teammate to the safe room. They evolved three independent neuro-controllers, then combined them using additional evolutionary runs. Lee [102] followed a similar idea for a box-pushing robot; Mouret et al. [132] combined this kind of incremental strategy with a MOEA to evolve a simulated flapping wing robot with heading control.

Fitness shaping Fitness shaping is an alternative to incremental evolution that also intends to guide the evolutionary process with sub-goals. Instead of switching fitness functions, fitness shaping consists of designing a single fitness function that rewards intermediate behaviors. For instance, Nolfi et al. evolved a neuro-controller for a robot that had to navigate in an arena, pick up objects and drop them outside of the arena [138]. Nolfi et al. designed a fitness function that was increased each time the robot had an object in the gripper and each time it released an object outside of the arena. They managed to obtain working controllers because they inserted a "hint" in the fitness function.

Fitness shaping can be damaging for the evolutionary process when the hint is misleading: because it aggregates the rewards for all the sub-behaviors in a single function, fitness shaping imposes to obtain the maximum reward for all the sub-goal, and therefore changes the optimum of the fitness function by imposing. Hence, if a simple behavior exists that was not foreseen by the experimenter, this behavior may be sub-optimal according to the fitness function, whereas it might solve the target task in a more efficient way than the one that obtains the best possible fitness score. While the intent of the authors was to design a process helper, fitness shaping could therefore be considered as a task-specific goal refiner. Another important issue with fitness shaping is that combining all the sub-goals in a single fitness function requires weighting them, which is often difficult without many preliminary experiments.

Multi-objective evolutionary algorithms have been exploited to overcome both of these issues – deceptive hints and weighting – to lead to "multi-objective fitness shaping" [127]. When each sub-goal is an objective in a Pareto-based multi-objective evolutionary algorithm, the Pareto-optimal set is made of the best individual for each sub-task, but also of all the other Pareto-optimal trade-offs, that is, of all the individuals that have a unique combination of partial skills. In this situation, a Pareto-based multi-objective evolutionary algorithm will therefore simultaneously optimize all the potentially fruitful combinations of sub-tasks to reach the goal task. This Pareto-based approach does not impose any order for the sub-tasks, nor make every sub-task mandatory or require weighting each of them. Mouret and Doncieux [128] illustrated these features with a light-switching task in which a simulated mobile robot had to switch lights on in a predefined order to switch on a "goal light". Two sequences were possible and one involved a shortest path to the goal light. Multi-objective fitness shaping allowed Mouret and Doncieux to successfully find working neuro-controllers following the shortest path. A similar multi-objective fitness shaping approach was recently employed to evolve the gait of a humanoid robot [143].

The knowledge used to shape a fitness function can be automatically extracted from a previous experiment on a simpler version of the task [49]. Two different approaches were tested in variants of the ball collecting tasks: a multi-objective one in which an objective was dedicated to the shaping and an approach in which the shaping term was aggregated to the fitness with a sum. Both approaches lead to similar results [49]. This corresponds to a more agnostic version of fitness shaping than that was previously mentioned, but as it implies to consider different versions of the same task, it still requires some knowledge about the task and can be considered a task-specific approach.

4.2.2 Task agnostic

Competitive co-evolution When two species compete, for instance predators and prey, each species change the selection pressure on the other one [39]. This situation may give rises to an "evolutionary arms race" in which each population drives the other to increasing levels of complexity. From the selective pressure point of view, such arm races correspond to a self-regulating adaptation of the task difficulty.

Many authors have investigated variants of the "predator-prey" task [77, 63, 140, 64, 65, 29, 148, 28, 139], sometimes called "pursuit-evasion" [122, 32, 33, 136]. In this task, a population of Khepera robots, the prey, interact by tournaments with other Khepera robots, the predators. Prey and predator robots differ: prey are faster than predators but only predators have a vision system. In the original studies, only the weights of the neuro-controllers were evolved. In a follow-up study [29, 28], the features of the sensor systems of both prey and predators could also be modified by the evolutionary algorithm. Other authors investigated similar tasks in which Khepera-like robots compete, for instance a game of "capture the flag" [135], a "duel" in which each robot must collect more energy than its opponent and collide with it [172], or a simplified soccer game [148].

Results of these experiments indicate that co-evolution can produce a never-ending evolution of strategies and counter-strategies, but that the co-evolutionary process easily enters a limit cycle in which the same strategies are abandoned and rediscovered over and over again [64, 139]. Stanley and Mikkulainen [172, 170], however, noted that this issue is mitigated when evolving neuro-controllers with an encoding that progressively complexifies networks, in this case NEAT [171].

The "red queen effect"[6] [181] is another classic issue with experiments in co-evolution [32, 63]: because the fitness landscape is ever changing, tracking progress is difficult, and so is avoiding retrogression.

It may be challenging to formulate all evolutionary robotics tasks in the form of a predator-prey task. An alternative is to replace the predator-prey co-evolution by "candidate solution"-"test case" co-evolution [82, 41]. In this case, the success of each robot (or robot controller) in the population is measured as their success with regard to the population of test cases. This population of test-cases is co-evolved and the success of each test case depends on how many robots fail the test: a test that is failed by all the candidate solutions is too hard, and one that is passed by all of them is too easy. This approach has been used several times to improve simulators. For instance, Bongard et al. employed model-test co-evolution to find a simulator of their quadruped robot [25, 13, 26] and Nardi et al. [44] to find an accurate simulator of a quadrotor; Koos et al. [96] extended this approach to Pareto co-evolution [41] and also applied it to a simulated quadrotor. Berlanga et al. [9, 10] used a similar strategy to increase the generalization abilities of neuro-controllers evolved in a Khepera navigation task. In a similar setup, Sakamoto and Zhao [158] compared co-evolution to incremental evolution; they concluded that co-evolution can be better than incremental evolution.

Behavioral diversity The most popular approach to mitigating premature convergence in evolutionary computation is undoubtedly to foster the diversity of the population [68, 159, 117, 67, 42, 174, 171]. Diversity is typically encouraged in the genotype space, but such an approach is computationally expensive for many genotypes used in evolutionary robotics and, in particular, for neural networks whose topology is evolved [171, 131]. NEAT [171] introduces a computationally cheap mechanism to encourage diversity in the genotypic space when evolving neural networks, but this technique does not solve the general problem of computing the similarity of two weighted graphs. Encouraging diversity in the space of genotypes ensures a good exploration of this space, but another space is actually involved in fitness evaluation: the space of behaviors.

An alternative line of thought has thus emerged during the last few years: whatever is evolved, the goal in evolutionary robotics is ultimately to find a *behavior* [104, 175, 176, 126, 129, 130, 73, 124, 104–106, 110, 51, 131, 52, 177]. This behavior results from the interaction of the robot with its environment, and is thus influenced by the robot controller—whether it is a neural network or anything else—and its morphology—whether it is evolved or not. By comparing behaviors instead of genotypes or phenotypes, the previously

[6]The term "red queen effect" is a reference to a statement made by the Red Queen to Alice In Lewis Carrol's Through the Looking-Glass [30]: "Now, *here*, you see, it takes all the running you can do, to keep in the same place."

	why?					how?							
	Fitness definition	Reducing evaluations	Generalization	Premature convergence	Reality gap	Shaping	Multi-objective EA	Staged	Mono-objective EA	Evaluation conditions	*Ad hoc* EA	Co-evolution	Interactive EA
Task-specific													
Behavioral decomposition [55, 102, 180, 111, 45]				•				•	•				
Behavioural memory [40]				•				•	•	•			
Concurent layered learning [183]				•				•		•		•	
Environmental complexification [74]				•				•		•		•	
Fitness shaping [138]				•		•			•				
Incremental MOEA [8]				•			•	•		•			
Incremental evolution [81, 184, 151, 7, 6]				•				•	•	•			
Layered learning [87]				•				•	•	•			
MO-ER [125]	•					•	•						
Multi-objective shaping [127]				•		•	•						
SGOCE [58, 95, 94]				•				•	•	•			
Scaffolding [14, 5, 4, 19, 18]				•				•	•	•			
Semi-interactive evolution [31, 22]				•			•						•
Shaping with transfer [49]				•		•	•	•					
Staged MOEA [132]				•			•	•					
Task-agnostic													
Behavior-based speciation [175–177]				•							•		
Behavioral diversity [129, 143, 51, 124, 52, 131, 145]				•			•						
Co-evolution environment/controllers [158, 10, 9]			•									•	
Competitive co-evolution [33, 28, 172, 77, 29, 65, 122, 136, 139, 32, 170, 140, 178, 62, 64, 63, 135, 148]				•								•	
Diversity w. behavioral distance [73]				•					•				
Minimal criteria NS [105]				•					•				
Novelty search (NS) [70, 104, 72, 106, 108, 100]				•					•				
Novelty-based multiobjectivization [126, 156, 110]				•			•						
Novelty-fitness aggregation [36, 69]									•				
Prog. minimal criteria NS [71]				•					•				
Two-population novelty search [112]	•			•					•				

Table 2. Process helpers. Each line corresponds to an article or set of articles about a similar topic (with respect to selective pressures). The first five columns describe the addressed challenges and the remaining ones the way they have been addressed.

described problems of comparing structures disappear and the diversity of behavior can be explicitly encouraged.

Behavioral diversity techniques rely on a behavioral distance, that is, a way to compare the behavior of individuals given one or several simulations. The first studies used Euclidean distances between task-specific behavior descriptors [129, 130]. Gomez [73] binarized the sensory-motor stream, which is more generic than task-specific descriptors, and compared streams using the normalized compression distance. In a ball-collecting task, Doncieux and Mouret [51] tested three generic behavior similarity measures, all based on the sensory-motor stream (Hamming distance, Fourier distance, state count) and a distance based on the trajectory of the robot in the arena; the Hamming distance led to the best result and is quick to compute. Gomes [69] successfully used a similar measure in two swarm robotics domains. Last, Doncieux and Mouret proposed to randomly switch between behavioral similarity measures, so that there is no need to choose between similarity measures; they benchmarked this approach in a ball collecting-task and in a hexapod locomotion task and concluded that this "dynamic similarity measure" improves the overall efficiency of behavioral diversity [52].

In a large benchmark, Mouret and Doncieux [131] investigated how to improve behavioral diversity in three different tasks (deceptive maze [106, 104, 110], sequential light switching [127], and ball collecting [51, 145, 154, 52]), with two different encodings (weight encoding and graph-based direct encoding), three different diversity mechanisms methods (multi-objective diversity [42, 174, 130, 129], fitness sharing [68, 159, 117], and multi-objective fitness sharing), and three different distances (genotypic distance, task-specific distance and distance based on the sensory-motor stream). They concluded that, no matter the encoding or the task, (1) encouraging diversity helps mitigate premature convergence, (2) multi-objective diversity mechanisms outperform fitness sharing, (3) behavioral diversity outperforms genotypic diversity, (4) the generic stream-based distance can be as effective as a task-specific distance. These conclusions are consistent with those of other authors who have investigated humanoid locomotion [143], a behavioral distance to replace the genotypic distance in NEAT [175–177, 124], and behavioral crowding in the tartarus domain [73].

Novelty search When the fitness function is highly deceptive [68], it may misdirect the search process towards dead-ends instead of guiding it [104, 106]. Lehman and Stanley argue that, in these cases at least, it may be useful to get rid of the fitness function and only search for novel behaviors. They also argue that this radical departure from objective-based search may be a better abstraction of how natural evolution continuously "discovers" new lifeforms. Lehman and Stanley exploited this idea in "novelty search" [104, 108, 106], an evolutionary process in which solutions are compared according to their behavior, like in behavioral diversity, and are ranked according to their novelty with regard to all the behaviors that have been discovered before. Two properties of evolutionary robotics experiments make novelty search different from exhaustive search. First, many genotypes lead to the same behavior [131]; therefore finding novel behaviors is very different, and potentially more efficient, than finding novel genotypes, which would be closer to exhaustive search. Second, an encoding like NEAT [171] starts with simple neural networks and progressively complexifies them. As a consequence, looking for novel behaviors with NEAT implies the exploration of simple behaviors first.

Lehman and Stanley first illustrated this technique in the "deceptive maze" domain, in which a simulated mobile robot has to take a large detour to reach the end of a small maze [104, 126]. They subsequently showed that novelty search is an effective algorithm to find neuro-controllers for bipedal walking [106]. Other authors have confirmed that novelty search is a promising alternative to objective-based search. For instance, Risi et al. showed that novelty search is capable of circumventing the deceptive trap of "learning to learn" when evolving plastic neural networks [156], Krčah successfully applied it to body-brain co-evolution [100], and Gomes et al. to swarm robotics [71].

Novelty search highlights that black-box objective-based search is not the only possible abstraction of evolution in evolutionary robotics. However, as noted by several authors [36, 105, 110, 126], novelty search does not scale well to spaces in which there are many possible behaviors, in particular because the assumption that many genotypes will lead to the same behavior does not hold any more. One way to mitigate this issue is to change the behavior distance (see the previous section); another one is to combine novelty and objective-based search (see the next section). Lehman and Stanley proposed a third idea, called "minimal criteria novelty search" [105], in which individuals must meet domain-dependent criteria to be selected for reproduction. They concluded that this approach can lead to better results than simple novelty search in a deceptive maze navigation task. This idea has been extended by Gomes et al. [71], who proposed to make the criteria dependent on the success of the current population. They illustrated their approach with a swarm robotics task [71].

Combining novelty and objective-based search Novelty search can be combined with objective-based search thanks to multi-objective evolutionary algorithms: one objective is the traditional fitness function, the other the novelty score used in novelty search. In effect, this approach is a variant of multi-objective behavioral diversity in which diversity is computed thanks to an archive in addition to the population [126, 161]. Mouret investigated such a "novelty-based multi-objectivization" in the same deceptive maze as Lehman and Stanley [126]. He concluded that the multiobjectivization is better at fine-tuning behaviors than basic novelty search while requiring a comparable number of iterations to converge; the novelty-based multi-objectivization was also faster to converge than basic behavioral diversity. Lehman and Stanley studied the efficiency of a similar multi-objectivization when the task difficulty increases. They compared it to age-based diversity [161], novelty alone, fitness alone, fitness & NEAT-like speciation, novelty & NEAT-like speciation and novelty+age, in the bipedal locomotion domain and in variants of the deceptive maze navigation task. They showed that, as difficulty increases, methods including a novelty objective perform better than methods without such an objective. Moreover, in the bipedal domain, the most effective approach is optimizing novelty and fitness together.

An alternative to multi-objective optimization is to combine a novelty and a fitness objective using a weighted sum. Cuccu and Gomez tested this approach in the Tartarus domain [36]; Gomes et al. used it in swarm robotics [69]. In both cases, the authors report that combining fitness and novelty worked better than novelty or fitness alone. Compared with novelty-based multiobjectivization, an obvious limitation of these techniques is that they require balancing the weight given to the novelty objective. Cuccu and Gomez, for instance, report large variations in performance when they changed this weight [36].

Semi-interactive evolution When an evolutionary process is trapped in a local optimum, a sensible option is to ask for help from the user. Celis et al. [31] investigated this idea in a system in which the user demonstrates what he or she prefers, in a quadruped robot locomotion and in an obstacle avoidance task. Using a similar task, Bongard and Hornby also introduced a multi-objective approach in which a surrogate user (which stands for the user) deflects the search away from local optima and a traditional fitness function leads the search toward the global optimum [22].

5 Conclusion and discussion

Selective pressures are of an increasing importance in ER. We have reviewed this literature while separating techniques aimed at changing final solution features (goal refiners) and techniques aimed at increasing search efficiency (process helpers). We have further divided these categories into two sub-categories: task-specific and task-agnostic approaches.

The main lessons from this review can be summarized as follows:

1. selection pressures have a critical importance in ER, as exemplified by the work on novelty search [106] and behavioral diversity [131], for instance;
2. including task-specific knowledge does not necessarily help [106];
3. taking into account exploration in the space of behaviors is important [131, 177, 106, 73, 104, 129];
4. multi-objective approaches offer a convenient way to combine selective pressures [131, 129, 127, 126, 110];
5. changing morphological or environmental complexity helps [16, 19, 18, 14].

Overall, this literature review shows that interesting results can be generated when ER is not considered purely as *black-box* optimization. The question can be put a bit differently: in retrospect, is ER an optimization problem? What questions are raised by this perspective? Are there alternatives? What questions do they raise? All these questions concern what the evolutionary process looks for and then the selective pressures that will favor some solutions at the expense of others. They suggest future work on ER.

Considering the evolutionary design process as an optimization problem requires precisely defining the goal, the associated fitness function(s), and the evaluation processes. If "intelligent behaviors" are to be generated, how should intelligence be defined and how it be tested? What about efficient locomotion, manipulation or perception skills? What tasks should be considered? Besides the definition of these goals and related evaluations *per se*, the difficulty of reaching it through an evolutionary search raises other questions. Is it possible to generate such complex systems without going through specific stepping stones? For intelligent systems endowed with cognitive abilities, counting may be a stepping stone [80], as well as having a memory [147] or internal representations [144]. These questions suggest a trend of research in which, for a goal recognized to be important, e.g., a robot with significant cognitive, motor or perception abilities, the corresponding stepping stones are proposed and studied. An alternative would be to study the main features of stepping stones in the hope of finding discriminative non-functional properties, like the evolvability they confer to the process for instance. It would open new perspectives as looking explicitly for these properties would be less task-specific and would then be closer to the goal of building an automated design process.

From an optimization perspective, it is common practice to define and use benchmarks to compare different approaches. Benchmarks are problems on which there is a common agreement that solving them more efficiently means that a significant progress has been made. In the context of ER seen as an optimization process, what would be the benchmarks to use?

A second point of view is possible: considering the evolutionary search process as an open-ended creative process. Instead of finding the best design to perform a particular task, a creative process aims at exploring original designs. In this case, a run will be successful if designs with a "good" potential are found and goes on to be found as long as the process is not stopped. It raises new questions: how do we drive a creative process? What criteria do we use for the selection process? How do we make it open-ended? Novelty search [106] is a possible starting point, but it raises a new question: in what space do we measure novelty? Another approach is to define and formalize interestingness and define a curiosity-driven search process [160, 149] or to rely on more indirect selective pressures as in co-evolution [172, 64, 65] or environment-driven pressures [27]. Considering the search process as a creative process also raises another important question: on what scientific method do we rely? For research on optimization, normal science [101] relies on benchmarks. Can we define benchmarks for a creative process? Is this a good idea? If not, what alternative can we follow for normal science in this field? Proof-of-concepts, in which new and innovative approaches are shown, are useful to demonstrate potential, but a more rigorous and formalized definition is required to permit comparisons.

A historical perspective on ER and on the use of prior knowledge shows an encouraging trend. The very first studies on evolutionary robotics included a lot of prior knowledge in the fitness function or in the evaluation. The hypothesis was that including knowledge helps, no matter what knowledge is included and how. A task-agnostic approach was then assumed to be less effective than a task specific approach. The experimenter chose the level of knowledge to be included as a trade-off between efficiency and desired autonomy. Nelson et al. have reviewed the literature on ER with this point of view [134]. Recent work shows a more complex picture. Including knowledge can be completely misleading [106], and must therefore be handled with care. When properly done, the gain may be significant [19, 17, 18, 4, 5, 146, 147], but more and more task agnostic methods have been proposed that also have a significant impact on ER efficiency [79, 72, 103, 131, 177, 106]. These new techniques are very encouraging and allows us to think about a future with a truly automated behavior design method.

6 Acknowledgements

This work has been funded by the ANR Creadapt project (ANR-12-JS03-0009)

References

[1] E. Alpaydin. *Introduction to machine learning*. The MIT Press, 2004.

[2] P. J. Angeline. Competitive fitness evaluation. In D.B. Fogel T. Back and Z. Michalewicz, editors, *Evolutionary Computation 2*, pages 12–14. Taylor & Francis, 2000.

[3] J. E. Auerbach and J. C. Bongard. How robot morphology and training order affect the learning of multiple behaviors. In *Proc. of the IEEE Congress on Evolutionary Computation (CEC'2009)*, pages 39–46, 2009.

[4] J. E. Auerbach and J. C. Bongard. On the relationship between environmental and mechanical complexity in evolved robots. In *Proc. of Artificial Life conference (ALife XIII)*, pages 309–316, 2012.

[5] J. E. Auerbach and J. C. Bongard. On the relationship between environmental and morphological complexity in evolved robots. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'12)*, pages 521–528, New York, New York, USA, 2012. ACM Press.

[6] D. Bajaj and M. Ang. An Incremental Approach in Evolving Robot Behavior. *Proc. of the International Conference on Control, Automation, Robotics and Vision (ICARCV'2000)*, 2000.

[7] R. Barate and A. Manzanera. Evolution of visual controllers for obstacle avoidance in mobile robotics. *Evolutionary Intelligence*, 2(3):85–102, October 2009.

[8] G. J. Barlow, C. K. Oh, and E. Grant. Incremental evolution of autonomous controllers for unmanned aerial vehicles using multi-objective genetic programming. *Proc. of IEEE Conference on Cybernetics and Intelligent Systems (CIS'2004)*, 2:689–694, 2004.

[9] A. Berlanga, A. Sanchis, P. Isasi, and J. M. Molina. A General Learning Co-Evolution Method to Generalize Autonomous Robot Navigation Behavior. In *Proc. of the IEEE Congress on Evolutionary Computation (CEC'2000)*, pages 769–776, 2000.

[10] A. Berlanga, A. Sanchis, P. Isasi, and J. M. Molina. Neural Network Controller against Environment : A Coevolutive approach to Generalize Robot Navigation Behavior. *Journal of Intelligent and Robotic Systems*, 33(2):139–166, 2002.

[11] P. Blanchard, R. L. Devaney, and G. R. Hall. *Differential Equations*. Thompson, London, 2006.

[12] A. Boeing and T. Braunl. Leveraging multiple simulators for crossing the reality gap. In *Proc. of International Conference on Control, Automation, Robotics & Vision (ICARV'2012)*, pages 1113–1119, 2012.

[13] J. C. Bongard. Action-selection and crossover strategies for self-modeling machines. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'07)*, pages 198–205. ACM Press, 2007.

[14] J. C. Bongard. Behavior Chaining : Incremental Behavior Integration for Evolutionary Robotics. In *Proc. of Artificial Life Conference (ALife XI)*, pages 64–71, 2008.

[15] J. C. Bongard. Accelerating Self-Modeling in Cooperative Robot Teams. *IEEE Transactions on Evolutionary Computation*, 13(2):321–332, April 2009.

[16] J. C. Bongard. The utility of evolving simulated robot morphology increases with task complexity for object manipulation. *Artificial life*, 16(3):201–23, January 2010.

[17] J. C. Bongard. Innocent until proven guilty: Reducing robot shaping from polynomial to linear time. *IEEE Transactions on Evolutionary Computation*, 15(4):571–585, 2011.

[18] J. C. Bongard. Morphological and environmental scaffolding synergize when evolving robot controllers. In *Proc. of the international conference on Genetic and Evolutionary Computation Conference (GECCO'11)*, pages 179–186, 2011.

[19] J. C. Bongard. Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences (PNAS)*, 108(4):1234–1239, January 2011.

[20] J. C. Bongard. Evolutionary Robotics. *Communications of the ACM*, 56(08):74–83, 2013.

[21] J. C. Bongard and G. S. Hornby. Guarding Against Premature Convergence while Accelerating Evolutionary Search. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'10)*, pages 111–118. ACM, 2010.

[22] J. C. Bongard and G. S. Hornby. Combining fitness-based search and user modeling in evolutionary robotics. *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'13)*, pages 159–166, 2013.

[23] J. C. Bongard and H. Lipson. Automated damage diagnosis and recovery for remote robotics. *Proc. of the International Conference of Robotics and Automation (ICRA'2004)*, 4:3545–3550, 2004.

[24] J. C. Bongard and H Lipson. Automated robot function recovery after unanticipated failure or environmental change using a minimum of hardware trials. *Proc. of Evolvable Hardware*, pages 169–176, 2004.

[25] J. C. Bongard and H. Lipson. Once More Unto the Breach : Co-evolving a robot and its simulator. *Proc. of the International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*, pages 57–62, 2004.

[26] J. C. Bongard, V. Zykov, and H. Lipson. Resilient Machines Through Continuous Self-Modeling. *Science*, 314(5802):1118–1121, 2006.

[27] N. Bredeche and J.M. Montanier. Environment-driven Embodied Evolution in a Population of Autonomous Agents. *Parallel Problem Solving from Nature (PPSN XI)*, PPSN 6239:290—-299, 2010.

[28] G. Buason, N. Bergfeldt, and T. Ziemke. Brains, Bodies, and Beyond: Competitive Co-Evolution of Robot Controllers, Morphologies and Environments. *Genetic Programming and Evolvable Machines*, 6(1):25–51, March 2005.

[29] G. Buason and T. Ziemke. Competitive co-evolution of predator and prey sensory-motor systems. *Applications of Evolutionary Computing*, pages 605–615, 2003.

[30] L. Carroll. *Alice's Adventures in Wonderland AND Through The Looking Glass*. MacMillan, 1866.

[31] S. Celis, G. S. Hornby, and J. C. Bongard. Avoiding Local Optima with User Demonstrations and Low-level Control. In *Proc. of the IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 3403–3410, 2013.

[32] D. Cliff and G. F. Miller. Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations. In *Proceedings of the Third European Conference on Artificial Life*, pages 200–218. (LCNS volume 929), 1995.

[33] D. Cliff and G. F. Miller. Co-evolution of Pursuit and Evasion II : Simulation Methods and Results. In *Proc. of the International Conference on Simulation of adaptive behavior (SAB'96)*, 1996.

[34] J. Clune and H. Lipson. Evolving Three-Dimensional Objects with a Generative Encoding Inspired by Developmental Biology. In *Proc. of the European Conference on Artificial Life (ECAL'11)*, 2011.

[35] J. Clune, K. O. Stanley, R. T. Pennock, and C. Ofria. On the Performance of Indirect Encoding Across the Continuum of Regularity. *IEEE Transaction On Evolutionary Computation*, 15(3):346–367, 2011.

[36] G. Cuccu and F. Gomez. When novelty is not enough. In *Applications of Evolutionary Computation*, pages 234–243, 2011.

[37] A. Cully and J.-B. Mouret. Behavioral Repertoire Learning in Robotics. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'13)*, pages 175–182, 2013.

[38] R. Dawkins. *The Selfish Gene*. Oxford, 1976.

[39] R. Dawkins and J. R. Krebs. Arms Races between and within Species. *Proceedings of the Royal Society B: Biological Sciences*, 205(1161):489–511, September 1979.

[40] H. De Garis. Building Nanobrains with Genetically Programmed Neural Networks Modules. In *Proc. of the International Joint Conference on Neural Networks (IJCNN'1990)*, pages 511–516, 1990.

[41] E. D. De Jong and J. B. Pollack. Ideal Evaluation from Coevolution. *Evolutionary Computation*, 12(2):159–192, 2004.

[42] E. D. De Jong, R. A. Watson, and J. B. Pollack. Reducing bloat and promoting diversity using multi-objective methods. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'01)*, pages 11–18. ACM, 2001.

[43] K. A. De Jong. *Evolutionary computation: a unified approach*, volume 262041944. MIT press Cambridge, 2006.

[44] R. de Nardi and O.E. Holland. Coevolutionary modelling of a miniature rotorcraft. In *Proc. of the International Conference on Intelligent Autonomous Systems (IAS10)*, 2008.

[45] R. de Nardi, J. Togelius, O. E. Holland, and S. M. Lucas. Evolution of Neural Networks for Helicopter Control: Why Modularity Matters. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2006)*, pages 1799–1806. Ieee, 2006.

[46] K. Deb. *Multi-objectives optimization using evolutionnary algorithms*. Wiley, 2001.

[47] P. Delarboulas, M. Schoenauer, and M. Sebag. Open-Ended Evolutionary Robotics: an Information Theoretic Approach. In *Proc. of Parallel Problem Solving From Nature (PPSN XI)*, number 216342, pages 334–343, 2010.

[48] E. Di Mario, I. Navarro, and A. Martinoli. The Effect of the Environment in the Synthesis of Robotic Controllers: A Case Study in Multi-Robot Obstacle Avoidance using Distributed Particle Swarm Optimization. *Advances in Artificial Life, ECAL 2013*, pages 561–568, September 2013.

[49] S. Doncieux. Transfer Learning for Direct Policy Search: A Reward Shaping Approach. In *Proc. of the IEEE Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob 2013)*, 2013.

[50] S. Doncieux and J.-A. Meyer. Evolving Modular Neural Networks to Solve Challenging Control Problems. In *Proc. of the Fourth International ICSC Symposium on engineering of intelligent systems (EIS 2004)*, 2004.

[51] S. Doncieux and J.-B. Mouret. Behavioral diversity measures for Evolutionary Robotics. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2010)*, pages 1303–1310, 2010.

[52] S. Doncieux and J.-B. Mouret. Behavioral diversity with multiple behavioral distances. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2013)*, pages 1427–1434. Ieee, June 2013.

[53] S. Doncieux, J.-B. Mouret, N. Bredeche, and V. Padois. *Evolutionary Robotics: Exploring New Horizons*, pages 3–25. Springer, 2011.

[54] G. Dozier. Evolving robot behavior via interactive evolutionary computation: From real-world to simulation. In *Proceedings of the ACM Symposium on Applied Computing (SAC'2001)*, pages 340–344. ACM, 2001.

[55] M. Duarte, S. Oliveira, and A. L. Christensen. Hierarchical Evolution of Robotic Controllers for Complex Tasks. In *Proc. of the IEEE Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob 2012)*, 2012.

[56] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing (Natural Computing Series)*. Springer, 2008.

[57] A. Farchy, S. Barrett, P. MacAlpine, and P. Stone. Humanoid robots learning to walk faster: From the real world to simulation and back. In *Proc. of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'2013)*, pages 39–46, 2013.

[58] D. Filliat, J. Kodjabachian, and J.-A. Meyer. Evolution of neural controllers for locomotion and obstacle-avoidance in a 6-legged robot. *Connection Science*, 11:223—-240, 1999.

[59] D. Floreano, P. Dürr, and C. Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, January 2008.

[60] D. Floreano and C. Mattiussi. *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. Intelligent Robotics and Autonomous Agents. MIT Press, 2008.

[61] D. Floreano and F. Mondada. Evolutionary neurocontrollers for autonomous mobile robots. *Neural Networks*, 11(7-8):1461–1478, 1998.

[62] D. Floreano and S. Nolfi. Adaptive behavior in competing co-evolving species. In *Proc. of the European Conference on Artificial Life (ECAL'97)*, pages 378–387, 1997.

[63] D. Floreano and S. Nolfi. God save the red queen! Competition in co-evolutionary robotics. In *Proc. of the 2nd Conference on Genetic Programming*, volume 5, 1997.

[64] D. Floreano, S. Nolfi, and F. Mondada. Competitive co-evolutionary robotics: From theory to practice. *Proc. of the international conference on Simulation of adaptive behavior (SABâĂŹ98)*, pages 515–524, 1998.

[65] D. Floreano, S. Nolfi, and F. Mondada. Co-evolution and ontogenetic change in competing robots. *Advances in the evolutionary synthesis of intelligent agents*, pages 273–306, 2001.

[66] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons Inc, New York - London - Sydney, 1966.

[67] T. Friedrich, P. S. Oliveto, D. Sudholt, and C. Witt. Theoretical analysis of diversity mechanisms for global exploration. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'08)*, pages 945–952. ACM, 2008.

[68] D. E. Goldberg. Simple genetic algorithms and the minimal, deceptive problem. In L. Davis, editor, *Genetic algorithms and simulated annealing*, pages 74–88. San Mato: Morgan Kaufman Publisher, 1987.

[69] J. Gomes and A. L. Christensen. Generic Behaviour Similarity Measures for Evolutionary Swarm Robotics. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ13)*, pages 199–206, 2013.

[70] J. Gomes, P. Urbano, and A. L. Christensen. Introducing Novelty Search in Evolutionary Swarm Robotics. In *Proc. of the International Conference on Swarm Intelligence (ANTS'2012)*, pages 85–96, 2012.

[71] J. Gomes, P. Urbano, and A. L. Christensen. Progressive Minimal Criteria Novelty Search. *Advances in Artificial Intelligence âĂŞ(IBERAMIA)*, pages 281–290, 2012.

[72] J. Gomes, P. Urbano, and A.L. Christensen. Evolution of swarm robotics systems with novelty search. *Swarm Intelligence*, 7(2-3):115–144, 2013.

[73] F. J. Gomez. Sustaining diversity using behavioral information distance. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ09)*, pages 113–120. ACM, 2009.

[74] F. J. Gomez and R. Miikkulainen. Incremental Evolution of Complex General Behavior. *Adaptive Behavior*, 5(3-4):317–342, 1997.

[75] F. J. Gomez and R. Miikkulainen. Transfer of Neuroevolved Controllers in Unstable Domains. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ04)*, pages 957–968, 2004.

[76] S. J. Gould and E. S. Vrba. Exaptation-a missing term in the science of form. *Paleobiology*, 8(1):4–15, 1982.

[77] J. Grefenstette and R. Daley. Methods for competitive and cooperative co-evolution. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, 1996.

[78] F. Gruau and K. Quatramaran. Cellular Encoding for Interactive Evolutionary Robotics. In *Proc. of European Conference on Artificial Life (ECAL'97)*, pages 368–377, 1997.

[79] E. Haasdijk, B. Weel, and A. Eiben. Right on the monee. *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ13)*, pages 207–214, 2013.

[80] C. Hartland, N. Bredeche, and M. Sebag. Memory-enhanced evolutionary robotics: the echo state network approach. In *Proc. of the IEEE Congress on Evolutionary Computation (CEC'2009)*, pages 2788–2795, 2009.

[81] I. Harvey, P. Husbands, and D. Cliff. Seeing the light: artificial evolution; real vision. In D Cliff, P Husbands, J.-A. Meyer, and S Wilson, editors, *Proc. of the International Conference on Simulation of adaptive behavior (SABâĂŹ94)*, pages 392–401. MIT Press/Bradford Books, 1994.

[82] W. D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena*, 42(1):228–234, 1990.

[83] J. H. Holland. *Adaptation in Natural and Artificial Systems*. MI: University of Michigan Press, 1975.

[84] G. S. Hornby. Steady-state ALPS for real-valued problems. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'09)*, pages 795–802, New York, New York, USA, 2009. ACM Press.

[85] G. S. Hornby and J. B. Pollack. Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3):223–246, 2002.

[86] G.S. Hornby. ALPS : The Age-Layered Population Structure for Reducing the Problem of Premature Convergence. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCO'06)*, pages 815–822, 2006.

[87] W. H. Hsu and S. M. Gustafson. Genetic Programming And Multi-agent Layered Learning By Reinforcements. In *Proc.*

of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ02), pages 764–771, 2002.

[88] N. Jakobi. Evolutionary Robotics and the Radical Envelope of Noise Hypothesis. *Adaptive Behavior*, 6(1):131–174, 1997.

[89] N. Jakobi, P. Husbands, and I. Harvey. Noise and the Reality Gap: The Use of Simulation in Evolutionary Robotics. *Lecture Notes in Computer Science*, 929:704–720, 1995.

[90] M. T. Jensen. Helper-Objectives: Using Multi-Objective Evolutionary Algorithms for Single-Objective Optimisation. *Journal of Mathematical Modelling and Algorithms*, 3(4):323–347, 2004.

[91] A. S. Klyubin, D. Polani, and C. L. Nehaniv. Empowerment: A universal agent-centric measure of control. In *Proc. of the IEEE Congress on Evolutionary Computation (CEC)*, pages 128–135, 2005.

[92] J. Knowles, Richard A Watson, and D. Corne. Reducing local optima in single-objective problems by multi-objectivization. In *Evolutionary Multi-Criterion Optimization*, pages 269–283. Springer, 2001.

[93] J. D. Knowles, R. A. Watson, and D. W. Corne. Reducing Local Optima in Single-Objective Problems by Multi-objectivization. *Proc. of First International Conference on Evolutionary Multi-Criterion Optimization*, 1993:268–282, 2001.

[94] J. Kodjabachian, C. Corne, and J.-A. Meyer. Evolution of a Robust Obstacle Avoidance Behavior in Khepera: A comparison of Incremental and Direct Strategies. *Robotics and Autonomous Systems*, 1999.

[95] J. Kodjabachian and J.-A. Meyer. Evolution and development of Neural Networks Controlling Locomotion, Gradient-Following, and Obstacle-Avoidance in Artificial Insects. *IEEE Transactions on Neural Networks*, 9:796–812, 1997.

[96] S. Koos, J.-B. Mouret, and S. Doncieux. Automatic system identification based on coevolution of models and tests. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2009)*, pages 560–567, 2009.

[97] S. Koos, J.-B. Mouret, and S. Doncieux. Crossing the Reality Gap in Evolutionary Robotics by Promoting Transferable Controllers. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ10)*, pages 119–126, 2010.

[98] S. Koos, J.-B. Mouret, and S. Doncieux. The Transferability Approach : Crossing the Reality Gap in Evolutionary Robotics. *IEEE Transaction on Evolutionary Computation*, 17(1):122–145, 2013.

[99] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, 1993.

[100] P. Krcah. Solving deceptive tasks in robot body-brain co-evolution by searching for behavioral novelty. In *Proc. of the International Conference on Intelligent Systems Design and Applications (ISDA'2010)*, pages 284–289, 2010.

[101] T. S. Kuhn. *The structure of scientific revolutions* . University of Chicago Press, 1962.

[102] W. Lee. Evolving complex robot behaviors. *Information Sciences*, 121(1-2):1–25, December 1999.

[103] J. Lehman, S. Risi, D. D. Ambrosio, and K. O. Stanley. Encouraging reactivity to create robust machines. *Adaptive Behavior*, 21:484–500, 2013.

[104] J. Lehman and K. O. Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *Proc. of Artificial Life Conference (ALife XI)*, pages 329–336, 2008.

[105] J. Lehman and K. O. Stanley. Revising the Evolutionary Computation Abstraction: Minimal Criteria Novelty Search. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ10)*, pages 103–110, 2010.

[106] J. Lehman and K. O. Stanley. Abandoning objectives: evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, January 2011.

[107] J. Lehman and K. O. Stanley. Evolving a Diversity of Creatures through Novelty Search and Local Competition. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ11)*, pages 211–218, 2011.

[108] J. Lehman and K. O. Stanley. Novelty search and the problem with objectives. *Genetic Programming Theory and Practice IX*, pages 37–56, 2011.

[109] J. Lehman and K. O. Stanley. Evolvability is inevitable: increasing evolvability without the pressure to adapt. *PloS one*, 8(4):e62186, January 2013.

[110] J. Lehman, K. O. Stanley, and R. Miikkulainen. Effective diversity maintenance in deceptive domains. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ13)*, pages 215–222, New York, New York, USA, 2013. ACM Press.

[111] M. A. Lewis, A. H. Fagg, and A. Solidum. Genetic programming approach to the construction of a neural network for control of a walking robot. *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'1992)*, pages 2618–2623, 1992.

[112] A Liapis, GN Yannakakis, and Julian Togelius. Enhancements to constrained novelty search: Two-population novelty search for generating game content. In *Proc. of of the International Conference on Genetic and Evolutionary Computation (GECCO'13)*, pages 343–350, 2013.

[113] H. Lipson. Evolutionary Robotics and Open-Ended Design Automation. *Biomimetics*, 17(9):129–155, May 2005.

[114] H. Lipson and J. B. Pollack. Automatic design and manufacture of robotic lifeforms. *Nature*, 406:974–978, 2000.

[115] H. H. Lund and O. Miglino. Evolving and breeding robots. *Evolutionary Robotics*, pages 192–210, 1998.

[116] H. H. Lund, O. Miglino, L. Pagliarini, A. Billard, and A. Ijspeert. Evolutionary robotics-a children's game. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ1998)*, pages 154–158. Ieee, 1998.

[117] S. W. Mahfoud. Niching Methods. In T. Bäck, D. B. Fogel, and Z. Michalewicz, editors, *Handbook of Evolutionary Computation*. Taylor & Francis, 1997.

[118] J.-A. Meyer and A. Guillot. Biologically-inspired robots. In O. Siciliano, B. and Khatib, editor, *Handbook of Robotics*, pages 1–38. Springer, 2008.

[119] J.-A. Meyer, A. Guillot, B. Girard, M. Khamassi, P. Pirim, and A. Berthoz. The Psikharpax project: Towards building an artificial rat. *Robotics and Autonomous Systems*, 50(4):211–223, 2005.

[120] J.-A.. Meyer and S. Wilson. Simulation of adaptive behavior in animats: Review and prospect. *Proc. of the International Conference on Simulation of adaptive behavior (SABâĂŹ91)*, pages 2–14, 1991.

[121] O. Miglino, H. H. Lund, and S. Nolfi. Evolving Mobile Robots in Simulated and Real Environments. *Artificial Life*, 2(4):417–434, 1995.

[122] G. F. Miller and D. Cliff. Protean behavior in dynamic games: Arguments for the co-evolution of pursuit-evasion tactics. In *Proc. of the International Conference on Simulation of adaptive behavior (SABâĂŹ94)*, pages 411–420. MIT Press, 1994.

[123] D. E. Moriarty and R. Miikkulainen. Forming Neural Networks through Efficient and Adaptive Coevolution. *Evolutionary Computation*, 5(4):373–399, 1997.

[124] H. Moriguchi and S. Honiden. Sustaining Behavioral Diversity in NEAT. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ10)*, pages 611–618. ACM, 2010.

[125] A. Moshaiov and A. Ashram-Wittenberg. Multi-objective Evolution of Robot Neuro-Controllers. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2009)*, pages 1093–1100, 2009.

[126] J.-B. Mouret. Novelty-based Multiobjectivization. In *New Horizons in Evolutionary Robotics: Extended contributions of the 2009 EvoDeRob Workshop*, pages 139–154. Springer, 2011.

[127] J.-B. Mouret and S. Doncieux. Incremental Evolution of Animats' Behaviors as a Multi-objective Optimization. In *Proc. of the International Conference on Simulation of adaptive behavior (SABâĂŹ08)*, volume 5040, pages 210–219. Springer, 2008.

[128] J.-B. Mouret and S. Doncieux. MENNAG: a modular, regular and hierarchical encoding for neural-networks based on attribute grammars. *Evolutionary Intelligence*, 1:187–207, 2008.

[129] J.-B. Mouret and S. Doncieux. Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2009)*, pages 1161–1168, 2009.

[130] J.-B. Mouret and S. Doncieux. Using Behavioral Exploration Objectives to Solve Deceptive Problems in Neuro-evolution. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ09)*, pages 627–634. ACM, 2009.

[131] J.-B. Mouret and S. Doncieux. Encouraging Behavioral Diversity in Evolutionary Robotics: An Empirical Study. *Evolutionary computation*, 20(1):91–133, August 2012.

[132] J.-B. Mouret, S. Doncieux, and J.-A. Meyer. Incremental evolution of target-following neuro-controllers for flapping-wing animats. In *Proc. of the International Conference on Simulation of adaptive behavior (SABâĂŹ06)*, pages 606–618, 2006.

[133] J.-B. Mouret, S. Koos, and S. Doncieux. Crossing the Reality Gap : a Short Introduction to the Transferability Approach. In *Proceedings of the ALIFE workshop "Evolution in Physical Systems"*, 2012.

[134] A. L. Nelson, G. J. Barlow, and L. Doitsidis. Fitness functions in evolutionary robotics: A survey and analysis. *Robotics and Autonomous Systems*, 57(4):345–370, April 2009.

[135] A. L. Nelson, E. Grant, and T. C. Henderson. Evolution of neural controllers for competitive game playing with teams of mobile robots. *Robotics and Autonomous Systems*, 46(3):135–150, March 2004.

[136] G. Nitschke. Co-evolution of cooperation in a pursuit evasion game. *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, 2:2037–2042, 2003.

[137] Y. Nojima, F. Kojima, and N. Kubota. Trajectory generation for human-friendly behavior of partner robot using fuzzy evaluating interactive genetic algorithm. In *Proc. of the IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for the New Millennium*, volume 1, pages 306–311. IEEE, 2003.

[138] S. Nolfi. Evolving non-trivial behaviors on real robots: A garbage collecting robot. *Robotics and Autonomous Systems*, 22(3-4):187–198, December 1997.

[139] S. Nolfi. Co-evolving predator and prey robots. *Adaptive Behavior*, 20(1):10–15, December 2011.

[140] S. Nolfi and D. Floreano. How Co-Evolution can Enhance the Adaptive Power of Artificial Evolution: Implications for Evolutionary Robotics. In *Proceedings of the First European Workshop on Evolutionary Robotics (EvoRobot98)*, pages 22–38, 1998.

[141] S. Nolfi and D. Floreano. *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. Bradford Book, 2001.

[142] M. A. C. Oliveira, S. Doncieux, J.-B. Mouret, and C. M. Peixoto dos Santos. Optimization of Humanoid Walking Controller : Crossing the Reality Gap. In *Proc. of the IEEE-RAS International Conference on Humanoid Robots (Humanoids'2013)*, 2013.

[143] M. A. C. Oliveira and C. P. Santos. Multi-objective parameter CPG optimization for gait generation of a quadruped robot considering behavioral diversity. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'2011)*, pages 2286–2291. Ieee, September 2011.

[144] C. Ollion. *Emergence of Internal Representations in Evolutionary Robotics: influence of multiple selective pressures*. PhD thesis, Pierre and Marie Curie University, 2013.

[145] C. Ollion and S. Doncieux. Why and how to measure exploration in behavioral space. *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ11)*, pages 267–274, 2011.

[146] C. Ollion and S. Doncieux. Towards Behavioral Consistency in Neuroevolution. In *Proc. of the International Conference on Simulation of Adaptive Behavior (SABâĂŹ12)*, pages 177–186, 2012.

[147] C. Ollion, T. Pinville, and S. Doncieux. With a little help from selection pressures: evolution of memory in robot controllers. In *Proc. of Artificial Life Conference (ALife XIII)*, pages 407–414, 2012.

[148] E. H. Ostergaard and H. H. Lund. Co-evolving Complex Robot Behavior. In *Evolvable Systems: From Biology to Hardware*, pages 308–319, 2003.

[149] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286, 2007.

[150] J. Paredis. Coevolutionary algorithms. In *Evolutionary Computation 2*, pages 224–238. Taylor & Francis, 2000.

[151] G. B. Parker. The Incremental Evolution of Gaits for Hexapod Robots. *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ01)*, pages 1114–1121, 2001.

[152] R. Pfeifer and J. C. Bongard. *How the Body Shapes the Way we Think*. MIT Press, 2006.

[153] R. Pfeifer, M. Lungarella, and F. Iida. Self-organization, embodiment, and biologically inspired robotics. *Science*, 318(5853):1088–93, November 2007.

[154] T. Pinville, S. Koos, J.-B. Mouret, and S. Doncieux. How to Promote Generalisation in Evolutionary Robotics : the ProGAb Approach Formalising the Generalisation Ability. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ11)*, pages 259–266, 2011.

[155] M. Prokopenko, V. Gerasimov, and I. Tanev. Evolving spatiotemporal coordination in a modular robotic system. *Proc. of the International Conference on Simulation of adaptive behavior (SABâĂŹ06)*, pages 558–569, 2006.

[156] S. Risi, S. D. Vanderbleek, C. E. Hughes, and K. O. Stanley. How novelty search escapes the deceptive trap of learning to learn. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ09)*, pages 153–160. ACM, 2009.

[157] R. M. Roberts. *Serendipity: Accidental Discoveries in Science*. Wiley Science Editions, 1989.

[158] K. Sakamoto and Q. Zhao. A Study on Generating Good Environment Patterns for Evolving Robot Navigators. In *Proc. of the IEEE International Conference on Systems, Man and Cybernetics*, volume 4, pages 3280–3285. IEEE, October 2006.

[159] B. Sareni and L. Krahenbuhl. Fitness sharing and niching methods revisited. *IEEE Transactions on Evolutionary Computation*, 2(3):97–106, 1998.

[160] T. Schaul, Y. Sun, D. Wierstra, F. Gomez, and J. Schmidhuber. Curiosity-Driven Optimization. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2011)*, pages 1343–1349, 2011.

[161] M. Schmidt and H. Lipson. Age-fitness pareto optimization. *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ10)*, pages 543–544, 2010.

[162] H.-P. Schwefel. *Numerische Optimierung von Computer Mod-*

ellen mittels der Evolutionsstrategie. Birkhäuser, 1977.

[163] J. Secretan, N. Beato, D. B. D'Ambrosio, A. Rodriguez, A. Campbell, J. T. Folsom-Kovarik, and K. O. Stanley. Picbreeder: A Case Study in Collaborative Evolutionary Exploration of Design Space. *Evolutionary computation*, 19(3):373–403, October 2011.

[164] O. Siciliano, B. and Khatib. *Handbook of Robotics*. Springer, 2008.

[165] K. Sims. Evolving virtual creatures. In *Proc. of SIGGRAPH '94*, pages 15–22, New York, New York, USA, 1994. ACM Press.

[166] T. Smith, P. Husbands, and M. O'Shea. Neutral networks in an evolutionary robotics search space. In *Proc. of the IEEE Congress on Evolutionary Computation (CECâĂŹ2001)*, volume 1, pages 136–143 vol. 1, 2001.

[167] V. Sperati, V. Trianni, and S. Nolfi. Evolving coordinated group behaviours through maximisation of mean mutual information. *Swarm Intelligence*, 2(2-4):73—-95, 2008.

[168] O. Sporns and M. Lungarella. Evolving coordinated behavior by maximizing information structure, 2006.

[169] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci. A Hypercube-Based Indirect Encoding for Evolving Large-Scale Neural Networks. *Artificial Life*, 15(2):185—-212, 2009.

[170] K. O. Stanley and R. Miikkulainen. Continual coevolution through complexification. *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ02)*, pages 113–120, 2002.

[171] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

[172] K. O. Stanley and R. Miikkulainen. Competitive Coevolution through Evolutionary Complexification. *Journal of Artificial Intelligence Research*, 21:63–100, 2004.

[173] H. Takagi. Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, 2001.

[174] A. Toffolo and E. Benini. Genetic diversity as an objective in multi-objective evolutionary algorithms. *Evolutionary Computation*, 11(2):151–167, 2003.

[175] L. Trujillo, G. Olague, E. Lutton, and F. F. De Vega. Behavior-based speciation for evolutionary robotics. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ08)*, pages 297–298, New York, New York, USA, 2008. ACM.

[176] L. Trujillo, G. Olague, E. Lutton, and F. F. De Vega. Discovering several robot behaviors through speciation. In *Application of Evolutionary Computing: 4th European Workshop on Bio-Inspired Heuristics for Design Automation*, pages 165–174. Springer, 2008.

[177] L. Trujillo, G. Olague, E. Lutton, F. Fernández de Vega, L. Dozal, and E. Clemente. Speciation in Behavioral Space for Evolutionary Robotics. *Journal of Intelligent & Robotic Systems*, 64(3):323–351, January 2011.

[178] E. Uchibe, M. Nakamura, and M. Asada. Cooperative and competitive behavior acquisition for mobile robots through co-evolution. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ99)*, pages 425–430, 1999.

[179] J. Urzelai and D. Floreano. Incremental Evolution with Minimal Resources. *Proceedings of IKW99*, pages 796–803, 1999.

[180] J. Urzelai, D. Floreano, M. Dorigo, and M. Colombetti. Incremental robot shaping. *Connection Science*, 10(3):341–360, 1998.

[181] L. v. Valen. Body Size and Numbers of Plants and Animals. *Evolution*, 27(1):27–35, 1973.

[182] R. A. Watson, S. G. Ficici, and J. B. Pollack. Embodied Evolution : Distributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems*, 39:1–18, 2002.

[183] S. Whiteson, N. Kohl, R. Miikkulainen, and P. Stone. Evolving soccer keepaway players through task decomposition. *Machine Learning*, 59(1):5–30, 2005.

[184] J. F. Winkeler and B. S. Manjunath. Incremental evolution in genetic programming. *Proc. Third Annual Conference on Genetic Programming*, pages 403–411, 1998.

[185] B. G. Woolley and K. O. Stanley. On the deleterious effects of a priori objectives on evolution and representation. In *Proc. of the International Conference on Genetic and Evolutionary Computation (GECCOâĂŹ11)*, pages 957–964, New York, New York, USA, 2011. ACM Press.

[186] B. G. Woolley and K. O. Stanley. A Novel Human-Computer Collaboration: Combining Novelty Search with Interactive Evolution. In *Proceedings of GECCO'2014*, pages 1–8, 2014.

[187] J. C. Zagal, J. Delpiano, and J. Ruiz-del Solar. Self-modeling in humanoid soccer robots. *Robotics and Autonomous Systems*, 57(8):819–827, July 2009.

[188] J. C. Zagal and J. Ruiz-del Solar. Combining Simulation and Reality in Evolutionary Robotics. *Journal of Intelligent and Robotic Systems*, 50(1):19–39, March 2007.

[189] J. C. Zagal, J. Ruiz-del Solar, and P. Vallejos. Back to Reality: Crossing the Reality Gap in Evolutionary Robotics. *Proc. of IAV*, 2004.